JavaScript 参考 - JavaScript | MDN (mozilla.org)

String.prototype.split()：用指定分隔符分割字符串

```
        String.prototype.split = function (separator, limit) {
            //如果没有提供分隔符，直接把字符串作为数组的一个元素并返回
            if (typeof separator === 'undefined') {
                return [this.valueOf()];
            }
            //保证limit是一个整数
            if (limit) {
                limit = Number.parseInt(limit);
            }
            //定义一个数组来缓存我们分割出来的元素
            let arr = [];

            let tmp = '';
            for (let n = 0; n < this.length; n++) {
                let match = this[n];
                //分隔符的长度是任意的--》需要拼接并判断是不是分隔符
                for (let j = n + 1; j < n + separator.length; j++) {
                    match += this[j];
                }
                //判断是不是分隔符
                if (match === separator) {
                    arr.push(tmp);
                    tmp = '';
                    //从分隔符的结束位置开始循环
                    n += separator.length - 1;
                    //进入下一次循环
                    continue;
                }
                //在遇到分隔符之前把字符缓存到tmp里面
                tmp += this[n];
            }
            //最后一个分隔符后面的tmp
            arr.push(tmp);
            return limit ? arr.slice(0, limit) : arr;
        }
```

```
        let str = ',abc,def,g,h,j,';
        console.log(str.split(','));
        console.log(str.split(',', 3));
        console.log(str.split('-'));
        console.log(str.split('ab'));
```

String.prototype.padEnd()：用指定字符从末尾填充

```
String.prototype.padEnd = function (targetLength, padString = ' ') {
            //逻辑的严谨性和健壮性
            if(padString === null || padString === '' || Number.isNaN(padString) ){
                return this.valueOf();
            }
            //填充到指定的目标长度
            //最大化填充用的字符串
            let padstr = '';
            for(let i=0; i < Math.ceil(targetLength / padString.length); i++){
                padstr += padString;
            }
            console.log(padstr);
            //从开始位置去截取需要的字符串
            let tmp = '';
            for(let i=0; i < targetLength - this.length; i++){
                tmp += padstr[i];
            }
            return this.valueOf() + tmp;
        }

        let str = 'abc';
        console.log(str.padEnd(2, '000')); //abc
        console.log(str.padEnd(4, '000')); //abc0
        console.log(str.padEnd(5, '000')); //abc00
        console.log(str.padEnd(7, '000')); //abc0000
```

String.prototype.charAt()获取指定字符在第几位

```

```

```
        String.prototype.mycharAt = function(index){
            var newStr = "";
            var index = index;
            var arr = [];
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
                return newStr[index];
        }
        var str = " hello world";
        console.log(str.mycharAt(20));
```

String.prototype.concat() 拼接字符串

```
String.prototype.myconcat = function(str){
            var newStr = "";
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
            return newStr + str;
        }
        var str1 = "hello";
        console.log(str1.myconcat(" hell"));
        String.prototype.charCodeAt()
```

String.prototype.startsWith() 获取字符串是否以…开始

```
String.prototype.mystartsWith = function(char){
                var newStr = "";
                var arr = [];
                for(var i = 0;i < this.length;i++){
                    newStr += this[i];
                }
                var firstStr = newStr[0];
                //一次判断，如果传参为单字符等于字符串最后一位
                if(firstStr == char){
                    return true;
```

```
                }
                //二次判断，如果传参为字符串，判断是否为字符串最后一个单词
                arr = newStr.split(" ");
                if(arr[0] == char){
                    return true;
                }
                return false;
        }
        var str = "blu one?";
        console.log(str.mystartsWith('blu'));
```

String.prototype.endsWith() 获取字符串是否以...结束

```
String.prototype.myendsWith = function(char){
                var newStr = "";
                var arr = [];
                for(var i = 0;i < this.length;i++){
                    newStr += this[i];
                }
                var lastStr = newStr[newStr.length - 1];
                //一次判断，如果传参为单字符等于字符串最后一位
                if(lastStr == char){
                    return true;
                }
                //二次判断，如果传参为字符串，判断是否为字符串最后一个单词
                arr = newStr.split(" ");
                if(arr[arr.length - 1] == char){
                    return true;
                }
                return false;
        }
        var str = "blu one?";
        console.log(str.myendsWith('one?'));

```

String.prototype.includes() 字符串是否包括...

```
String.prototype.myincludes = function(char){
                var newStr = "";
                var arr = [];
                for(var i = 0;i < this.length;i++){
                    newStr += this[i];
                }
                //一次判断，如果传参为单字符
                for(var i = 0;i < newStr.length;i++){
                    if(newStr[i] == char){
                        return true;
                    }
                }
                //二次判断，如果传参为字符串
                arr = newStr.split(" ");
                for(var i = 0;i < arr.length;i++){
                    if(arr[i] == char){
                        return true;
                    }
                }
                return false;
        }
        var str = "blu one";
        console.log(str.myincludes('blue'));
```

String.prototype.indexOf() 指定字符的下标

```
 String.prototype.myIndexOf = function(char){
            var newStr = "";
            var index = -1;
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
            for(var i = 0;i < newStr.length;i++){
                if(newStr[i] == char){
                    index = i;
                    break;
                }
```

```
            }
            return index;
        }
        var str = "123111222122";
        console.log(str.myIndexOf(1));
```

String.prototype.lastIndexOf() 指定字符最后出现的下标

```
String.prototype.mylastIndexOf = function(char){
            var newStr = "";
            var index;
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
            for(var i = 0;i < newStr.length;i++){
                if(newStr[i] == char){
                    index = i;
                }
            }
            return index;
        }
        var str = "123111222122";
        console.log(str.mylastIndexOf(1));
```

String.prototype.replace() 替换指定字符串

```
String.prototype.myreplace = function(num1,num2){
            var newStr = "";
            var returnStr = "";
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
            var arr = newStr.split(' ');
            for(var i = 0;i < arr.length;i++){
                if(arr[i].includes(num1)){
                    arr[i] = num2;
                    break;
```

```
                }
            }
            for(var i = 0;i < arr.length;i++){
                returnStr += arr[i] + " ";
            }
            return returnStr;
        }
        const str = 'The quick brown fox jumps over the lazy dog If the dog reacted, was it really lazy?';
        console.log(str.myreplace('dog','cat'));
```

String.prototype.replaceAll() 替换所有符合条件的字符

```
 String.prototype.myreplaceAll = function(num1,num2){
            var newStr = "";
            var returnStr = "";
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
            var arr = newStr.split(' ');
            for(var i = 0;i < arr.length;i++){
                if(arr[i].includes(num1)){
                    arr[i] = num2;
                }
                returnStr += arr[i] + " ";
            }
            return returnStr;
        }
        const str = 'The quick brown fox jumps over the lazy dog. If the dog reacted, was it really lazy?';
        console.log(str.myreplaceAll('dog','cat'));
```

String.prototype.substr() //截取字符串

```
 String.prototype.mysubstr = function(start,length){
            var newStr = "";
            var returnStr = "";
            for(var i = 0;i < this.length;i++){
```

```
            newStr += this[i];
        }
        //没长度有开始值
        if(typeof length == "undefined"){
            //正数
            if(start > 0){
                if(start >= newStr.length){
                    return "";
                }else{
                    for(var i = start;i < newStr.length;i++){
                        returnStr += newStr[i];
                    }
                    return returnStr;
                }
            //负数
            }else if(start < 0){

                if(Math.abs(start) > newStr.length){
                    return newStr;
                }else{
                    for(var i = newStr.length + start;i <newStr.length;i++){
                        returnStr += newStr[i];
                    }
                    return returnStr;
                }
            }
        }
        //有长度有开始位置
        else{
            //开始位置为正数
            if(start > 0){
                if(start <= newStr.length){
                    for(var i = start;i <= length;i++){
                        returnStr += newStr[i];
                    }
                    return returnStr;
                }else{
                    return "";
                }
            }else if(start < 0){
```

```
                    if(Math.abs(start) > newStr.length){
                        for(var i = 0;i < length;i++){
                            returnStr += newStr[i];
                        }
                        return returnStr;
                    }else{
                        var count = 0;
                        for(var i = newStr.length + start;i <=newStr.length;i++){
                            count++;
                            returnStr += newStr[i];
                            if(count == length){
                                return returnStr;
                            }
                        }
                    }
                }
            }
        }
        var str = "abcdefghij";
        console.log(str.mysubstr(20,2));
```

String.prototype.trim() 去除字符串所有空格

```
String.prototype.mytrim = function(str){
            var newStr = "";
            var returnStr = "";
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
            for(var i = 0;i < newStr.length;i++){
                if(newStr[i] == " "){
                    continue;
                }else{
                    returnStr += newStr[i];
                }
            }
            return returnStr;
        }
```

```
        var a = "abc  12 3   a";
        console.log(a.mytrim());
```

String.prototype.toUpperCase() 转换成大写字母

```
String.prototype.mytoUpperCase = function(){
            var newStr = "";
            var returnStr = "";
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
            var temp = 0;
            for(var i = 0;i < newStr.length;i++){
                if(newStr[i].charCodeAt() < 97 && newStr[i].charCodeAt() > 64){
                    temp = newStr[i].charCodeAt() + 32;
                    returnStr += String.fromCharCode(temp);
                }else{
                    returnStr += str[i];
                }
            }
            return returnStr;
        }
        var str = "ABC";
        console.log(str.mytoUpperCase());
```

String.prototype.toLowerCase() 转换成小写字母

```
String.prototype.mytoLowerCase = function(){
            var newStr = "";
            var returnStr = "";
            for(var i = 0;i < this.length;i++){
                newStr += this[i];
            }
            var temp = 0;
            for(var i = 0;i < newStr.length;i++){
                if(newStr[i].charCodeAt() > 96){
                    temp = newStr[i].charCodeAt() - 32;
```

```
                    returnStr += String.fromCharCode(temp);
                }else{
                    returnStr += str[i];
                }
            }
            return returnStr;
        }
        var str = "ab c111  ";
        console.log(str.mytoLowerCase());
```

String.prototype.repeat():

```
String.prototype.repeat = function (n) {
            let str = '';
            for (let m = 1; m <= n; m++) {
                str += this;
            }
            return str;
        }

        let str = ' abc ';
        console.log(str.repeat(3)); //abc abc abc
```

js判断字符串是否是回文的三种方法

```
function fn1(str){
    let strReverse = str.split('').reverse().join('')
    return str == strReverse ? true : false
}
```

```
function fn2(str){
    let len = str.length
    if(len == 0 || len == 1) return true
    if(str.charAt(0) != str.charAt(len-1)){
        return false
```

```
    }else{
        return fn2(str.substring(1, len-1))
    }
}
```

```
function fn3(str){
    let len = str.length
    for(let i = 0, j = len-1; i <= j; i++,j--){
        if(str[i] != str[j]){
            return false
        }
    }
    return true
}
```